```
//In_Cons6.cpp
//Slicing in Call by value
# include <iostream.h>

class B {
   public:
      int x;
      B(const B & b)
         { x = b.x; cout << " Copy Constructing B\n";}
      B(int x_val): x(x_val){}
      void Display() const
         { cout << "B::x = " << x << endl;}
};

class D: public B {
   public:
      int  y;
      D(const D & d):B(d)
         { y = d.y; cout << " Copy Constructing D\n";}
      D(int x_val, int y_val): B(x_val), y(y_val){}
      void Display() const
         { cout << "D::y = " << y << endl ;}
};
```

```
void Do_D_val(D old_d)
    { cout << " inside Do_D_val, "; old_d.Display();}
void Do_D_ref(const D & old_d)
    { cout << " inside Do_D_ref, "; old_d.Display();}
void Do_B_val(B old_b)
    { cout << " inside Do_B_val, "; old_b.Display();}
void Do_B_ref(const B & old_b)
    { cout << " inside Do_B_ref, "; old_b.Display();}

main()
{ D derive(10,20);

 cout << " calling Do_D_val\n";
 Do_D_val(derive);

 cout << "\n calling Do_D_ref\n";
 Do_D_ref(derive);

 cout << "\n calling Do_B_val\n";
 Do_B_val(derive);

 cout << "\n calling Do_B_ref\n";
 Do_B_ref(derive);
}
```

```
OUTPUT
------

calling Do_D_val
Copy Constructing B
Copy Constructing D
inside Do_D_val, D::y = 20

calling Do_D_ref
inside Do_D_ref, D::y = 20

calling Do_B_val
Copy Constructing B
inside Do_B_val, B::x = 10

calling Do_B_ref
inside Do_B_ref, B::x = 10
```